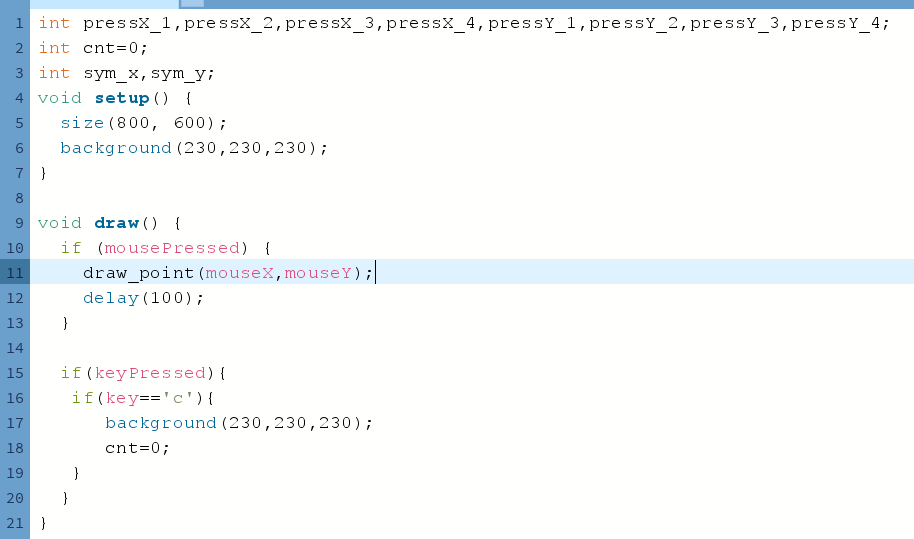
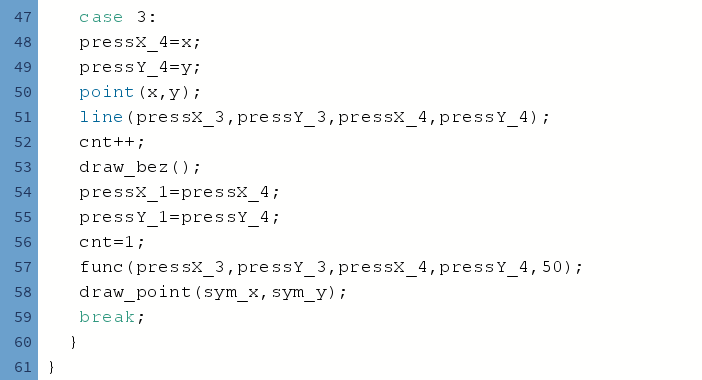
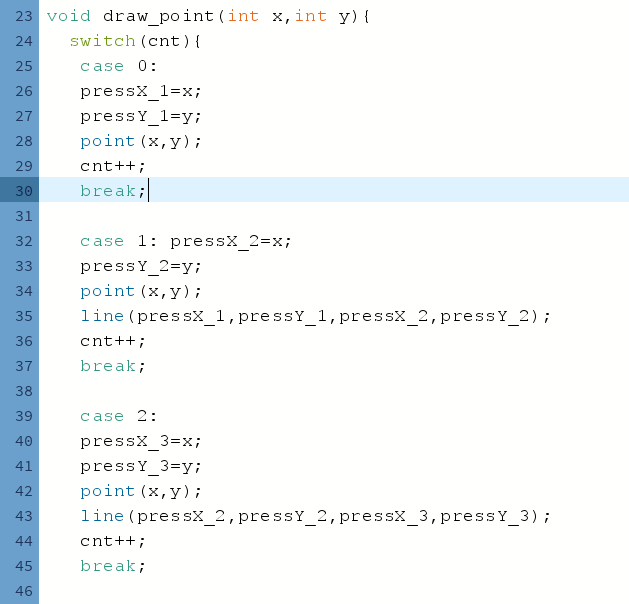
Алгоритм Без’є - параметрично задана крива, яка використовується в комп'ютерній графіці та суміжних галузях.

Ініціалізовуємо змінні для точок при натисканні мишки. Далі функція draw, у якій викликається функція draw\_point, яка при натисканні кнопки мишки відмальовує точку, також зробив своєрідне «очищення» вікна, насправді це просто заливка поверх тим самим кольором бекграунду.

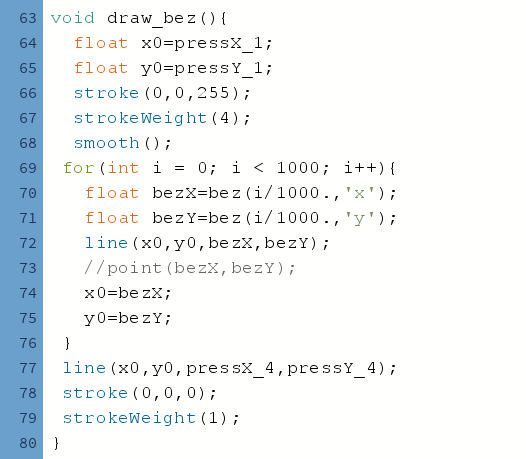


Далі власне моя функція draw\_point, яка відмальовує точку. Зробив я це через свіч кейс. Тобто в залежності від кількості точок на екрані я приймаю різні дії. В першому випадку це присвоюю своїм змінним х та у, будую точку, збільшую каунтер. Коли на екрані вже є одна точка, спрацьовує кейс 1: так само, при натисканні на екран моїм змінним присвоюються координати х та у, будується точка за цими координатами, а потім будується лінія між цією точкою та попередньою. Далі аналогічно.

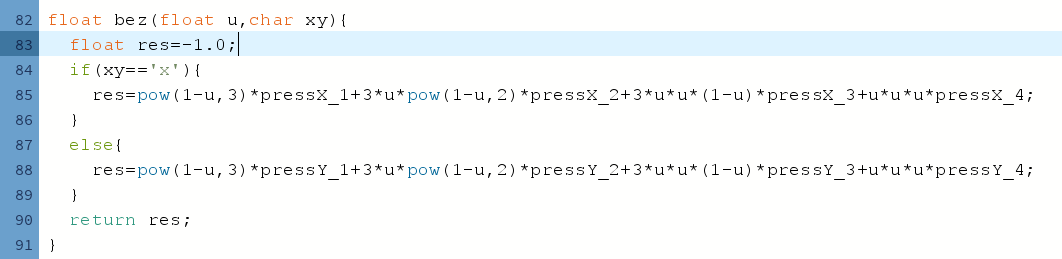
А вот потім, коли точок на екрані вже 3, спочатку приймаються аналогічні дії, тобто відмальовування точки і лінії, далі викликається функція draw\_bez(покажу чуть пізніше), переприсвоюються координати, а також викликається фукнція func і заново перевикликається функція draw\_point.



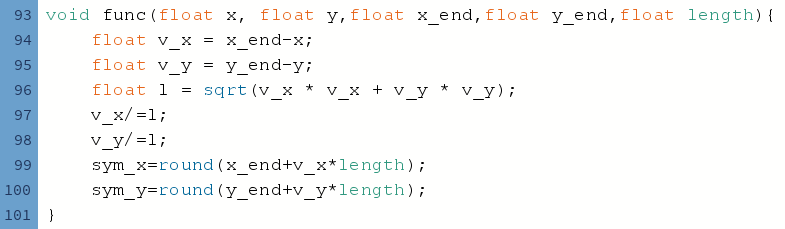
Далі функція draw\_bez, яку ви могли бачити у фукнції вище. Це функція, за домогою якої ми, власне, малюємо саму криву Без’є. Тут я хардкодом задав 1000 точок. Взагалі, я експериментував, і 1000 точок виявились найоптимальнішим варіантом, тут вони з’єднуються між собою, згладжуються і будують криву.



Далі функція bez, в якій я описав формули Без’є. Це формули для чотирьох контрольних точок.



Ну і функція func. В ній я обраховую довжину останньої прямої.



Існує і другий спосіб побудови кривих Без’є: це алгоритм де Кастельє:

